



RAPPORT DU DEVOIR

Graphe

Rapport du devoir d'algorithmique de graphes

COULON Anthony
ROLLAND Cyrille
09/11/2009

INDEX

INTRODUCTION	3
I. Implémentation d'un graphe	4
II. Les algorithmes	4
1/ Le DFS	4
2/ Kosaraju_Sharir	4
III. Mode d'emploi	5
IV. Les améliorations envisageables	13
CONCLUSION	14

INTRODUCTION

Dans le cadre des études d'ingénieur en deuxième année de l'école Sup Galilée à l'université paris 13, il nous a été demandé d'effectuer un projet d'algorithmique de graphes.

Ce projet à été entièrement codé par ROLLAND Cyrille et COULON Anthony sans utilisation de logiciel y compris pour la création de l'interface graphique. Nous avons cependant réutilisé certains codes vus en Interface Graphique durant notre première année d'étude afin de pouvoir créer les graphes sous formes graphique.

Ce projet consiste à réaliser un programme qui détermine le graphe réduit d'un graphe. Pour cela il nous est demandé d'utiliser l'algorithme de Kosaraju-Sharir ce qui implique donc d'implémenter l'algorithme du DFS.

Pour lancer le programme allez dans le répertoire « Sources » et tapez la commande « java ClassMain ». Il est préférable avant la première exécution d'exécuter le Makefile afin d'éviter de rencontrer des erreurs. Il est fortement conseillé de lire de « ReadMe ».

Nous étudierons dans ce rapport comment ces algorithmes ont été implémentés et nous fournirons une analyse des complexités. Dans une deuxième partie sera mis en place un petit mode d'emploi et pour finir nous verrons quelles sont les améliorations envisageables pour ce projet.

I. Implémentation d'un graphe :

Nous avons vu en TD que l'utilisation de matrice plutôt que celle de liste permet d'améliorer la plupart du temps la complexité des algorithmes de graphes.

Ceci est en particulier vrai pour :

- Déterminer si deux sommets sont adjacents.
- Déterminer les successeurs d'un sommet.
- Déterminer les prédécesseurs d'un sommet.

Ces fonctions étant régulièrement utilisées dans les algorithmes du DFS et donc Kosaraju_Sharir nous avons décidé d'utiliser une matrice pour représenter les arcs.

Afin de permettre à l'utilisateur de donner des valeurs à ces sommets, la classe Graphe contient également une liste de Sommets où le sommet à l'indice i de cette liste est la valeur donnée par l'utilisateur au sommet d'indice i .

II. Les algorithmes :

1/ Le DFS :

Pour le DFS, c'est l'algorithme vu en cours permettant d'avoir une liste des préfixes et des suffixes qui a été implémentée.

Il y a deux versions du DFS :

- Une version simple qui commence par le sommet d'indice 1.
- Une version où un tableau d'indice est passé en argument afin de donner un ordre au parcours de l'arbre. Cette deuxième version est donc utilisée lors du deuxième parcours DFS de l'algorithme de Kosaraju_Sharir.

Ces deux algorithmes étant quasi similaires, ils ont donc la même complexité. Elle correspond à celle vue en cours qui est en $O(m) + O(n)$ où m est le nombre d'arcs et n le nombre de sommets.

Ces algorithmes retournent un élément de la classe RéponseDFS qui se compose d'une liste de liste d'entiers correspondant à la forêt d'indices obtenue et un tableau d'entiers permettant de déterminer l'ordre des suffixes.

2/ Kosaraju Sharir :

L'algorithme pour Kosaraju_Sharir est également celui vu en cours, il s'agit simplement pour un graphe G :

- d'appliquer le DFS sur G .
- Construire G' qui est le graphe G où l'on a inversé le sens des arcs.
- Appliquer le DFS sur G' en démarrant par le sommet de plus grand numéro suffixe et itérer le processus à partir du sommet non marqué de plus grand numéro suffixe.

La complexité de cet algorithme est donc deux fois celle de DFS plus celle de la création du sous graphe qui se fait en $O(n^2)$ car il faut parcourir toute la matrice d'arcs pour pouvoir créer les arcs inversés.

III. Mode d'emploi :

Ce mode d'emploi est fait sous la forme d'un scénario explicatif :

- La première page, celle de bienvenue, de notre programme. A cette étape deux options s'offrent à vous, quitter en appuyant sur le bouton « Quitter » et entrer en appuyant sur le bouton « Entrer ». Pour lancer le programme et passer à la page suivante, il vous faut sélectionner « Entrer ».



Vous avez aussi à votre disposition un menu déroulant. Vous avez ainsi le choix dans « Menu » de sélectionner « Nouveau graphe » pour créer un nouveau graphe ou bien « Quitter » pour quitter le programme.



- Une fois que vous êtes entré dans le vif du sujet, vous pouvez maintenant commencer à créer votre graphe. Dans cette étape vous devez choisir combien de sommets votre graphe contient. Nous prendrons par exemple 5 sommets.

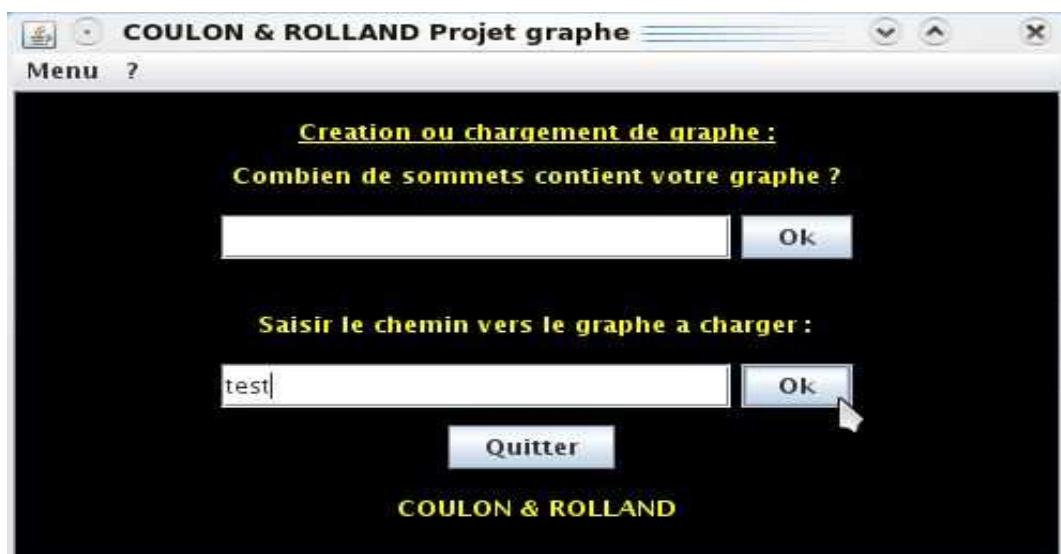
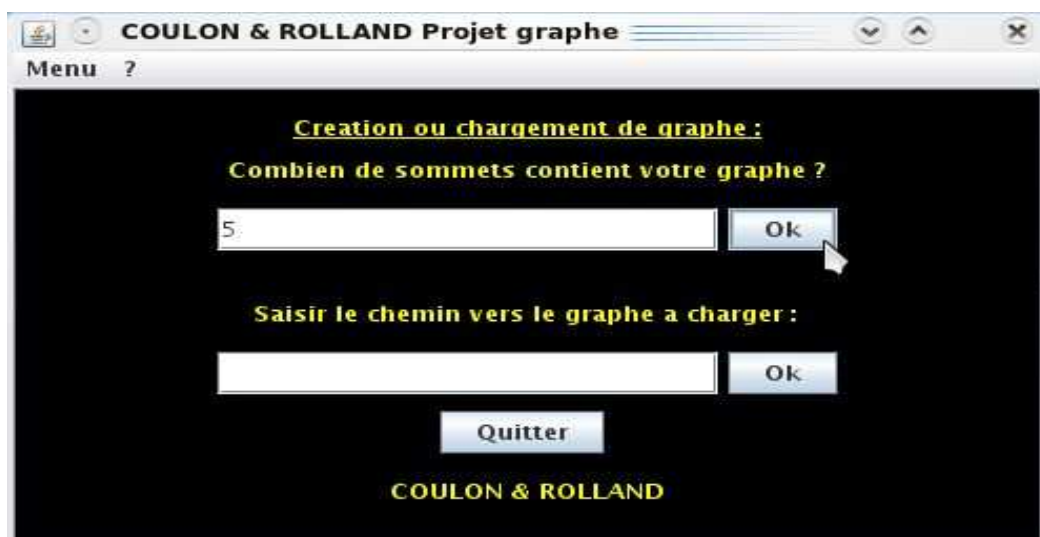
Vous avez la possibilité de charger un graphe déjà existant. Il vous suffit d'inscrire dans le champ nom du fichier contenant les informations du graphe que vous désirez charger. Vous arriverez directement à la page de résultat.

A savoir :

Les graphes que vous pouvez charger sont dans des fichiers « nomGraphe.g ».

Vous avez dans la page résultat la possibilité de générer un fichier texte décrivant votre graphe et les résultats obtenus.

Vous pouvez à ce moment du programme le quitter en appuyant sur le bouton « Quitter », sinon pour passer à la suite sélectionnez « Ok ».



- Vous avez la possibilité de donner des noms à vos sommets. Par défaut ils seront mis à zéro (par la suite on utilisera pour les sommets les noms ainsi que les indices). Pour notre exemple nous nommerons les sommets a, b, c, d et e. Pour passer à la suite nous appuierons sur le bouton « Suivant ».

COULON & ROLLAND Projet graphe

Menu ?

Valeur des sommets :

Donner les valeur de vos sommets
Par default elle vaudra 0

Sommet d'indice 1 :	a
Sommet d'indice 2 :	b
Sommet d'indice 3 :	c
Sommet d'indice 4 :	d
Sommet d'indice 5 :	e

Quitter **Suivant**

COULON & ROLLAND

- A cette étape on vous demande de créer les arcs. Vous devez indiquer les indices des sommets entre lesquels il y a un arc. Attention si un 0 est présent dans un arc alors cet arc ne sera pas construit. Un arc est construit deux fois, il sera construit et donc pris en compte qu'une seule fois. Pour notre exemple nous donnerons comme arc : 1->2, 1->3, 2->1, 2->3, 3->1, 4->5, 5->4. Pour passer à la phase de résultat vous devez appuyer sur « Calculer ».

COULON & ROLLAND Projet graphe

Menu ?

Creation des arcs :

Saisissez les arcs :

Indice des sommets:	
Source	Cible
1	2
1	3
2	1
2	3

ATTENTION :
-> Si un 0 est present dans votre arc
alors l'arc ne sera pas construit
-> Un arc saisie 2 fois ne sera cree
qu'une fois

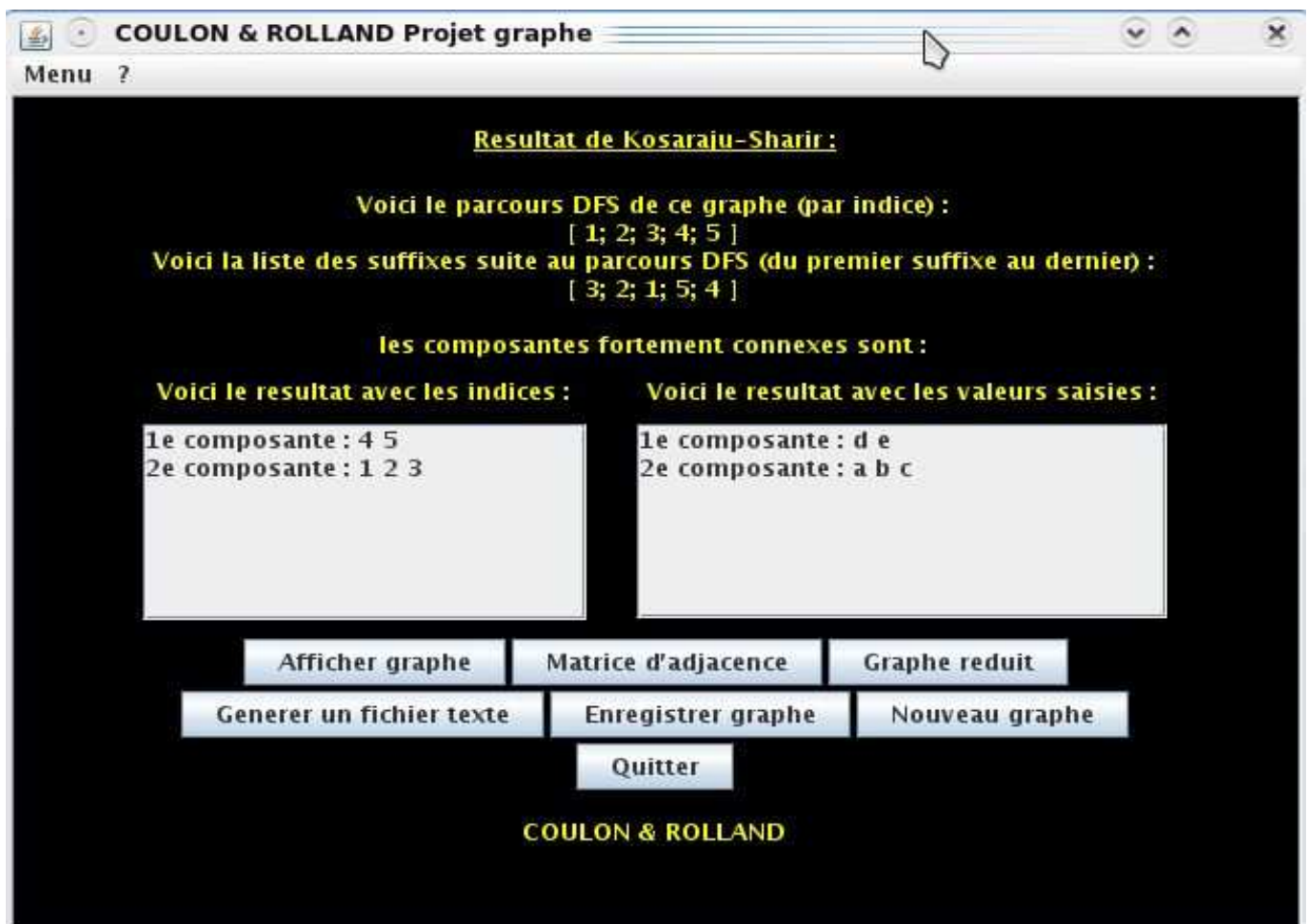
Quitter **Calculer**

COULON & ROLLAND

- Cette étape nous indique tous les résultats à partir de notre graphe. Nous avons les composantes fortement connexes obtenues avec l'algorithme de Kosaraju-Sharir, le parcours DFS ainsi que les suffixes suite au parcours DFS.

Pour les composantes fortement connexes nous avons deux résultats, un premier donné avec les indices et un deuxième avec les noms des sommets rentrés par l'utilisateur. Nous avons la possibilité :

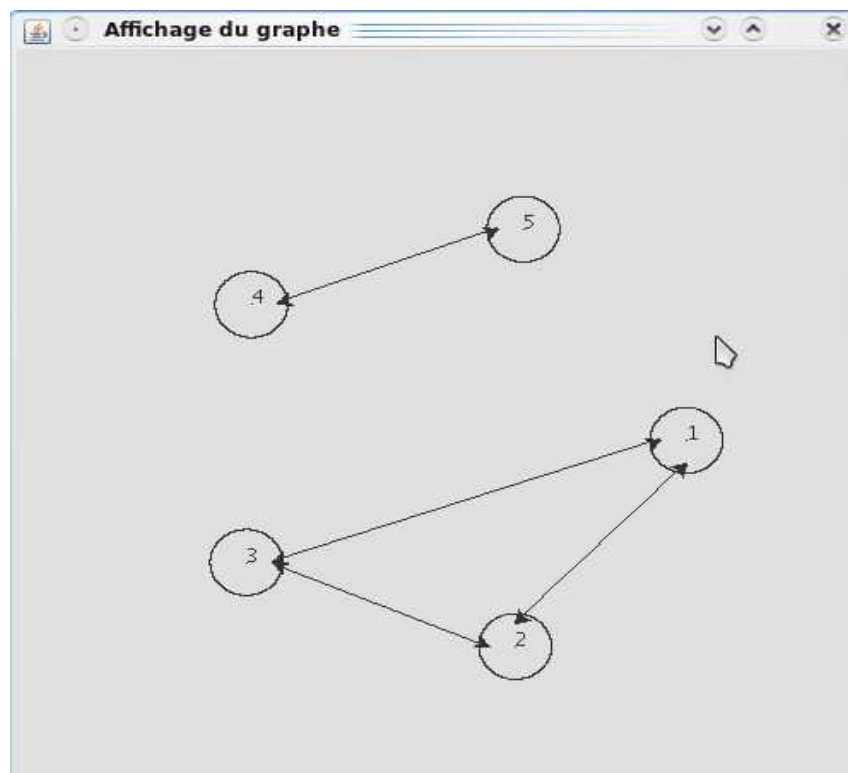
- d'afficher les informations sur le graphe réduit avec le bouton « Afficher graphe réduit »
- de générer un fichier texte avec le bouton « Générer un fichier texte »
- d'afficher la matrice d'adjacence avec le bouton « Matrice d'adjacence »
- de créer un nouveau graphe avec le bouton « Nouveau graphe »
- d'enregistrer le graphe avec le bouton « Enregistrer graphe »
- d'afficher le graphe avec le bouton « Afficher graphe »
- de créer un nouveau graphe avec le bouton « Nouveau graphe »
- de quitter avec le bouton « Quitter »



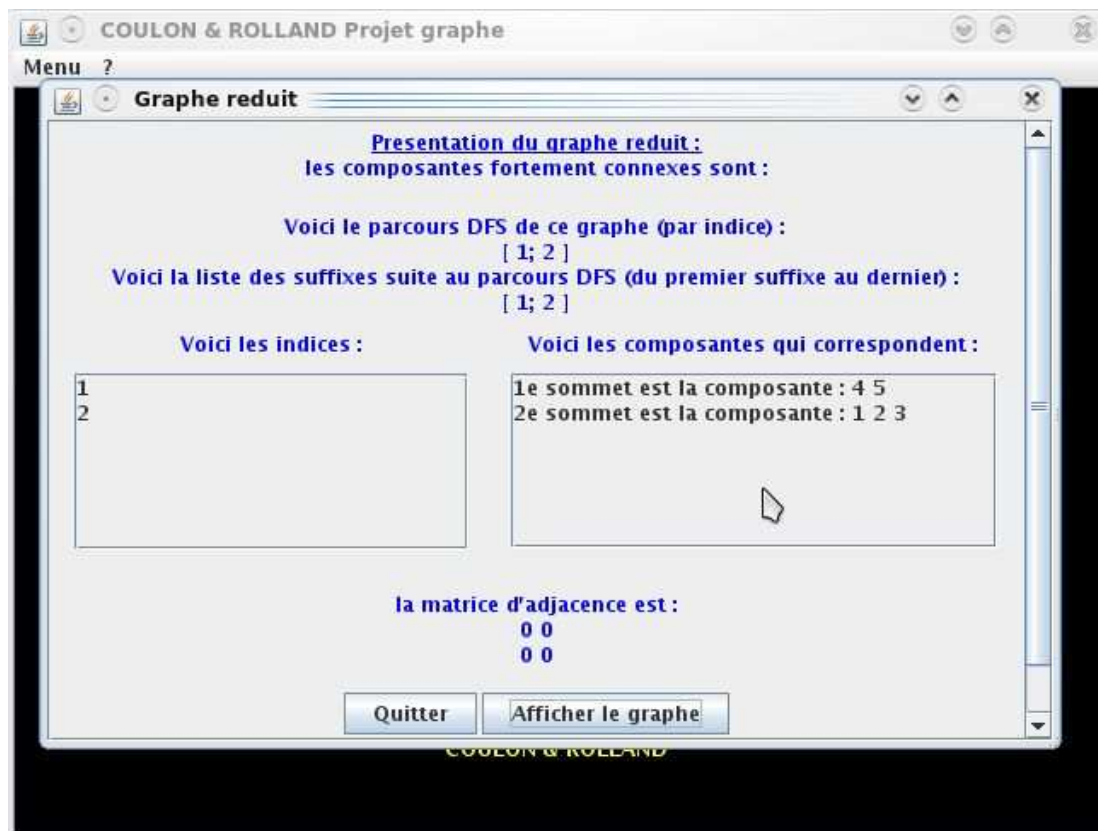
- o Afficher la matrice d'adjacence :



- o Après avoir appuyé sur le bouton « Afficher graphe » vous arrivez sur cette fenêtre :

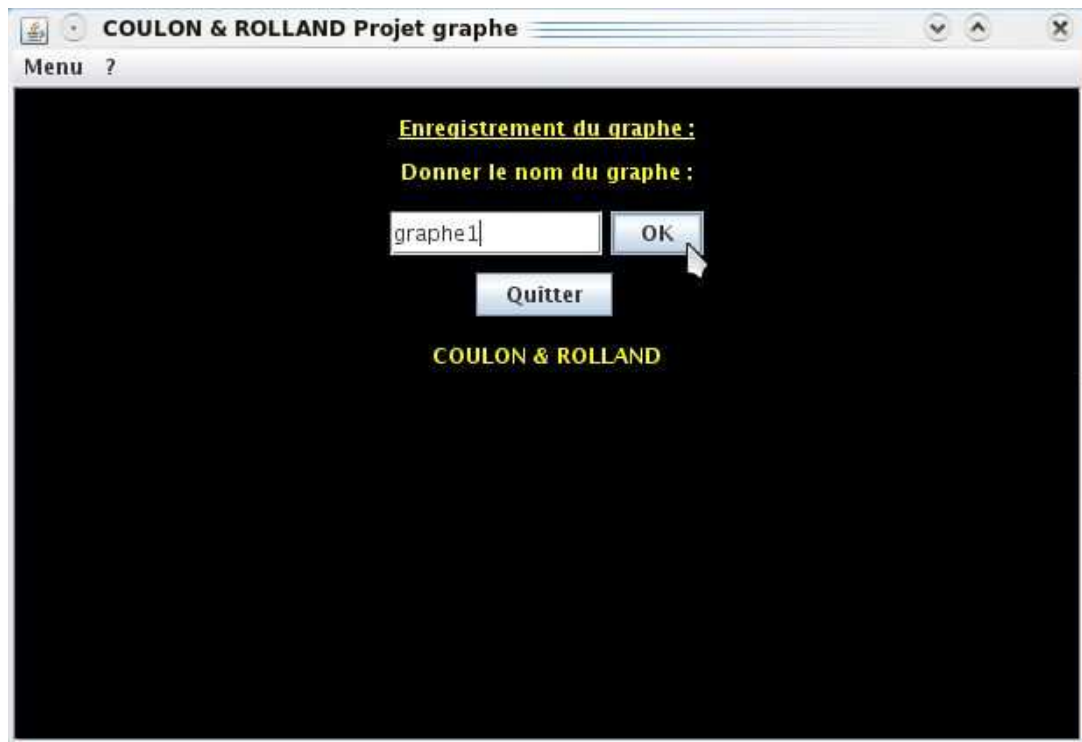


- o Nous avons ici les résultats concernant les graphes réduits. Cela nous indique les composantes fortement connexes. Nous avons les indices ainsi que les sommets qu'ils représentent. Nous avons ici le DFS et les suffixes du graphe réduit et la matrice d'adjacence, vous pouvez également afficher ce graphe.

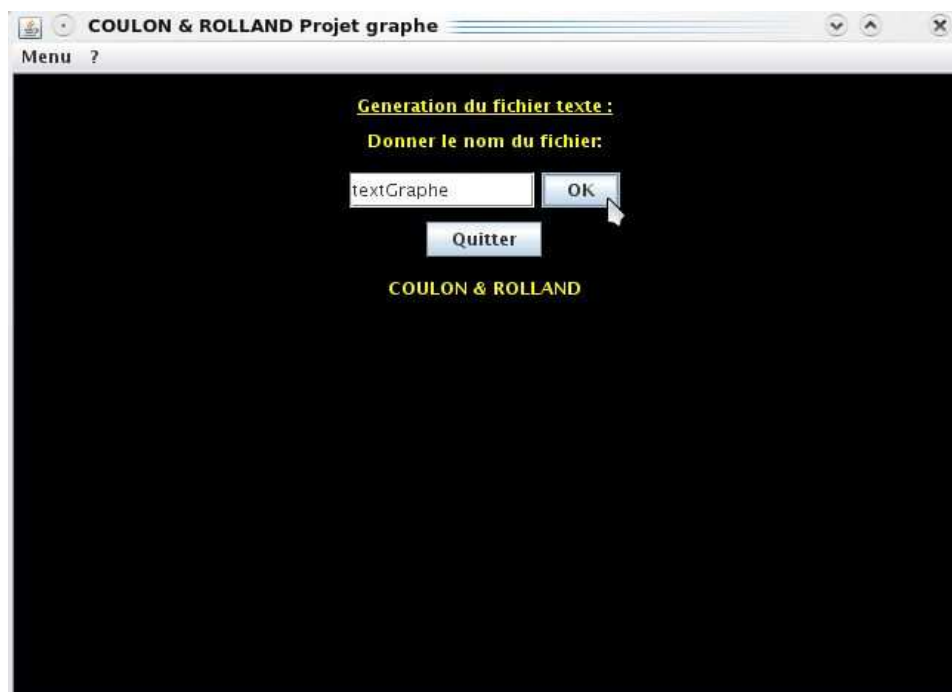


- Après avoir appuyé sur le bouton « Enregistrer graphe » vous arrivez à cette fenêtre. Vous indiquez le nom de votre graphe, par exemple « graphe1 ». Ensuite pour enregistrer il vous faut appuyer sur le bouton « Ok ». Sinon appuyer sur le bouton « Quitter » pour quitter.

A SAVOIR : Cliquer sur ok va créer un fichier graphe1.g



- Après avoir cliqué sur le bouton « Generer un fichier texte » vous arrivez à cette fenêtre. On vous demande de rentrer le nom de votre futur fichier texte qui contiendra les informations de votre graphe. Par exemple ici on donne pour nom « textGraphe ». L'extension est par défaut en .txt donc il n'est pas nécessaire de la préciser, le fichier en sortie sera donc « textGraphe.txt ».



Voici un exemple du contenu d'un fichier texte :

Voici les elements du graphe :

Ce graphe possede 5 sommets

Voici les valeurs de vos sommets :

Le sommet d'indice 1 a pour valeur 0

Le sommet d'indice 2 a pour valeur 0

Le sommet d'indice 3 a pour valeur 0

Le sommet d'indice 4 a pour valeur 0

Le sommet d'indice 5 a pour valeur 0

Le parcours DFS est (par indice): [1; 2; 3; 4; 5]

Les suffixes sont dans l'orde du plus petit au plus grand [3; 2; 1; 5; 4]

La matrice d'adjacence est :

0 1 1 0 0

1 0 1 0 0

1 1 0 0 0

0 0 0 0 1

0 0 0 1 0

Les composantes fortement connexes sont :

la 1e composante est : 45

la 2e composante est : 123

Le sous graphe est :

Graphe de 2 sommets{

Sommet d'indice 0 vaut 4 5

Sommet d'indice 1 vaut 1 2 3

}

La matrice d'adjacence de ce graphe est :

0 0

0 0

IV. Les améliorations envisageables :

Le temps étant limité pour faire ce projet il reste certaines améliorations qui pourraient être intéressantes à implémenter. Celles-ci sont :

- Amélioration de l'interface graphique
- Offrir à l'utilisateur la possibilité d'obtenir les résultats au format HTML
- Améliorer la qualité graphique des graphes
- Permettre la modification d'un graphe

CONCLUSION

L'implémentation de ce logiciel à été faite par COULON Anthony et ROLLAND Cyrille. Ce projet répond à l'ensemble des demandes du devoir. Le logiciel obtenu a une interface graphique qui le rend extrêmement simple d'utilisation.